

Graph based Version of KNN in Combining Text Summarization with Text Categorization

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—In this research, we propose the graph based KNN version for implementing the automatic text summarization system by introducing the text categorization. In the previous work, we proposed the KNN version as an approach to the text summarization, but must present the domain as a tag, manually. This research modifies the KNN algorithm into the graph-based version and applies it to the text summarization and the text classification. In the proposed system, only a text without its domain tag is given as the input, the domain is automatically assigned to the text by the text classification, and each paragraph is classified into summary or non-summary by the classifier which corresponds to the domain. The goals of this research are to automate completely the text summarization by combining it with the text classification, and to improve the performance of both tasks, using the graph based KNN.

I. INTRODUCTION

The text summarization is defined as the process of selecting and extracting the essential part from a text in context of the text mining. Because the essential part is assumed to be some among the paragraphs in the text, the text summarization is interpreted into a binary classification of paragraphs into summary or non-summary. The process of text summarization is to partition a text which is given as the input into paragraphs, classify paragraphs into one of the two categories, and extract ones which are classified into summary. The text summarization is mapped into a domain dependent classification where a same paragraph may be classified differently, depending on the domain. Because the text summarization is given as an independent binary classification domain by domain, a domain is required as a tag for nominating a corresponding binary classifier.

This research is motivated by the text summarization system which requires the domain as the additional input to the text. The text summarization is defined as an independent binary classification domain by domain; a machine learning-based classifier is allocated to each domain. We need to know the domain of the input text in advance for nominating the corresponding classifier. It is possible to automate the process of assigning a domain to a text before summarizing a text through the text classification. This research is intended to connect the text summarization to the text classification for solving the problem.

The idea of this research is to apply the graph based KNN algorithm to the text summarization which relates to the text classification. We define the similarity metric between graphs and modify the KNN algorithm into the version which processes graphs directly. The graph based KNN algorithm is adopted for implementing the text classification which assigns a domain to the input text. In the process of text summarization, a text is partitioned into paragraphs, each of them is classified into summary or non-summary by the modified KNN algorithm, and ones which are classified into summary are extracted. Only a text is presented without its own domain for executing the text summarization system.

Let us mention some benefits from this research. Encoding texts into graphs is the solution to the problems in encoding them into numerical vectors. The chance for modifying other machine learning algorithms into graph-based versions is opened by modifying the KNN algorithm so. The performance of the text summarization and the text classification is improved by adopting the graph based KNN algorithm as the approach to both tasks. We automate the process of deciding a domain for activating the text summarization system by connecting the text summarization with the text classification.

Let us mention remaining tasks for upgrading this research. We may define and characterize mathematically other operations in addition to the similarity metric between graphs. Paragraphs and texts are encoded into compound representations, each of which consists of more than one structured form. Other machine learning algorithms and deep learning algorithms are modified into graph-based versions. The text summarization system which relates to the text classification module is implemented by adopting the graph based KNN algorithm as a real program or a library.

II. PROPOSED WORKS

A. Encoding Proccerss

This section is concerned with the graph as a text representation and the similarity between graphs. In representing a text into a graph, a word is given as a vertex, and a similarity between words is given as an edge. A graph is viewed as a set of edges, and the similarity between edges

is computed before computing the similarity between graphs. The similarity between them is computed by averaging the similarities of all possible edge pairs. This section is intended to describe the process of encoding a text into a graph and the process of computing the similarity between graphs.

The process of encoding a text into a graph is illustrated in Figure 1. Words are retrieved as the vertices from the text by indexing it. The similarities among words are computed as edges in the edge definition. In the edge selection, the edges with their higher similarities are selected for representing a text into a graph. Refer to [1] for studying the encoding process in more detail.

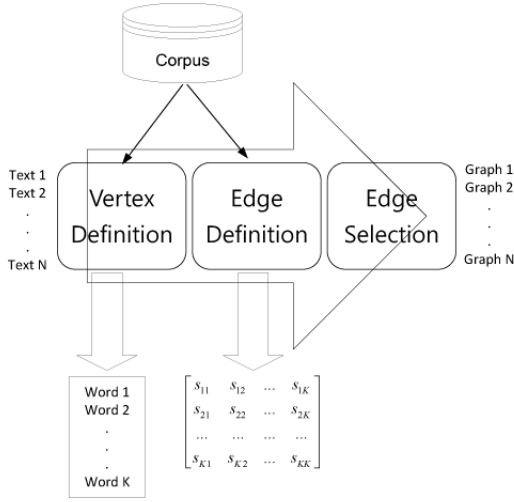


Figure 1. Process of Encoding Texts into Graphs

The three cases of computing the similarity between two edges are illustrated in Figure 2. Each weight which is associated with its own edge is assumed to be a normalized value between zero and one, and if both nodes are identical to each other, the average over two weights is the similarity between two edges. If either of the two nodes is identical to each other, the product of two weights is the similarity between two edges. If neither of two nodes is identical, zero is the similarity between them. The similarity between two edges is always a normalized value between zero and one.

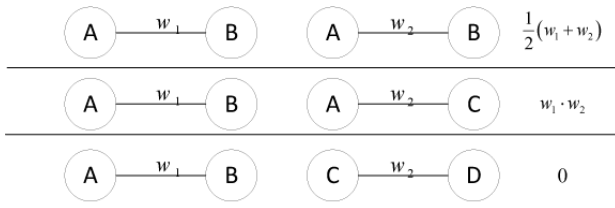


Figure 2. Three Cases of Comparing Two Edges with Each Other

Let us mention the process of computing the similarity between graphs as a semantic similarity between texts. The two

graphs which represent texts are notated by edge sets, $G_1 = \{e_{11}, e_{12}, \dots, e_{1|G_1|}\}$ and $G_2 = \{e_{21}, e_{22}, \dots, e_{2|G_2|}\}$. Two edges, $e_{1i} \in G_1$ and $e_{2j} \in G_2$, are associated with their own weights, w_{1i} and w_{2j} , and the similarity between two edges, e_{1i} and e_{2j} by equation (1), (2), or (3), depending on the cases which are mentioned above,

$$sim(e_{1i}, e_{2j}) = \frac{1}{2}(w_{1i} + w_{2j}) \quad (1)$$

$$sim(e_{1i}, e_{2j}) = w_{1i} \times w_{2j} \quad (2)$$

$$sim(e_{1i}, e_{2j}) = 0 \quad (3)$$

The similarity between two graphs is computed by equation (4),

$$sim(G_1, G_2) = \frac{1}{|G_1| \times |G_2|} \sum_{i=1}^{|G_1|} \sum_{j=1}^{|G_2|} sim(e_{1i}, e_{2j}). \quad (4)$$

The value which is generated from equation (4), is always given as a normalized value between zero and one as shown in equation (5),

$$0 \leq sim(G_1, G_2) \leq 1. \quad (5)$$

Let us make some remarks on the encoding process and the computation of the similarity between texts. A text is encoded into a graph by the vertex definition, the edge definition, and the edge weighting. The three cases are considered for computing the similarity between edges. The similarity between graphs is computed by equation (4). In the future research, we consider representing a text into multiple graphs.

B. Graph based KNN Algorithm

This section is concerned with the graph based KNN algorithm. It was initially proposed as the approach to the word classification by Jo in 2018 ???. The similarity metric between graphs which was described in the previous section is used for computing the similarity between a novice graph and a training example. The labels of its nearest neighbors are voted with their identical weights for deciding the label of a novice input. This section is intended to describe the initial version of KNN algorithm from which its variants are derived.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded into graphs, and they are notated by a set, $Tr = \{G_1, G_2, \dots, G_N\}$. A novice word is encoded into a graph notated by G . Its similarities with the graphs which represent the sample words are computed by equation (4), as $sim(G, G_1), sim(G, G_2), \dots, sim(G, G_N)$. Some training examples which are most similar as the novice input are selected as its nearest neighbors.

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples

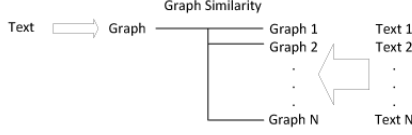


Figure 3. Similarities of Novice Input with Training Examples

are sorted by the descending order of their similarities, after computing their similarities with a novice example. The training examples with their higher similarities with a novice example are selected as its nearest neighbors, as expressed in equation (6),

$$Nr = \text{Select}_k(Tr, G), Nr \subseteq Tr \quad (6)$$

The set of nearest neighbors, Nr , is composed with elements as expressed in equation (7),

$$Nr = \{G_1^{near}, G_2^{near}, \dots, G_k^{near}\} \quad (7)$$

The elements in the set, Nr , are used for voting their labels in the next step.

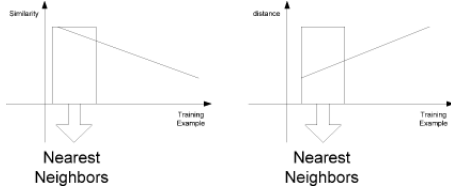


Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by $c_1, c_2, \dots, c_m$, and the label of a nearest neighbor is notated by $c_j = \text{label}(G_i^{near})$. The number of nearest neighbors which belong to the category, c_j , is notated by $\text{Count}(Nr, c_j)$. The label of the novice item, G , is decided by equation (8),

$$\text{label}(G) = \arg\max_{j=1}^m \text{Count}(Nr, c_j) \quad (8)$$

The process of deciding the label of a novice item by equation (8), is called voting.

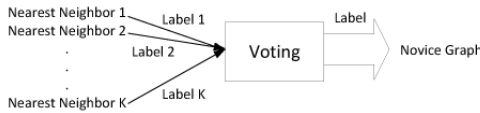


Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

Let us make some remarks on the graph based KNN algorithm. It computes the similarities of a novice item with the training examples. The training examples with their

highest similarities with the novice item are selected as its nearest neighbors. The label of the novice item is decided by voting the labels of its nearest neighbors. The ranking selection is adopted for selecting the nearest neighbors from training examples, in this version.

C. System Architecture

This section is concerned with the text summarization system which adopts the graph based KNN algorithm. In Section II-B, we described the proposed version of KNN algorithm as the approach to the text summarization. It is viewed as a binary classification which classifies a paragraph into summary or non-summary. This section is intended to describe the text summarization system with respect to its function and its architecture.

The process of sampling paragraphs domain by domain is illustrated in Figure 6. Even if it is possible map the text summarization into an instance of text classification, a paragraph may be classified differently depending on the domain. Sample paragraphs which are labeled with summary or non-summary are gathered domain by domain. The text which is given as the input is tagged with a domain, and the classifier which corresponds to the domain is appointed. The sample paragraphs are encoded into graphs in each domain.

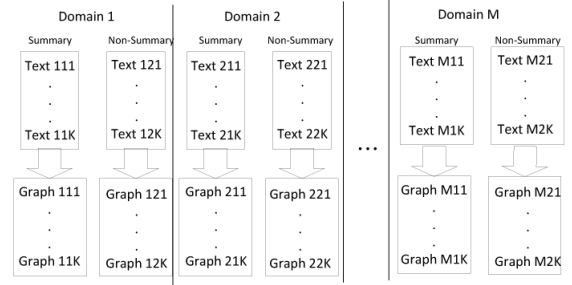


Figure 6. Preparation of Training Examples

The entire architecture of the proposed text summarization system is illustrated in Figure 7. A text is given as the input, and paragraphs are extracted by partitioning the text. In the encoding module, the sample paragraphs in the summary group and the non-summary group and ones which are extracted from the text are mapped into graphs in the encoding module. The paragraphs from the text are classified into one of the two categories in the similarity computation module and the voting module. The paragraphs which are classified into summary are extracted as the output in this system.

The execution flow of the text summarization system is illustrated in Figure 8. In each domain, sample paragraphs which are labeled with summary or non-summary are gathered. The input text is partitioned into novice paragraphs, and both sample paragraphs and novice paragraphs are encoded into graphs. Each paragraph is classified into summary

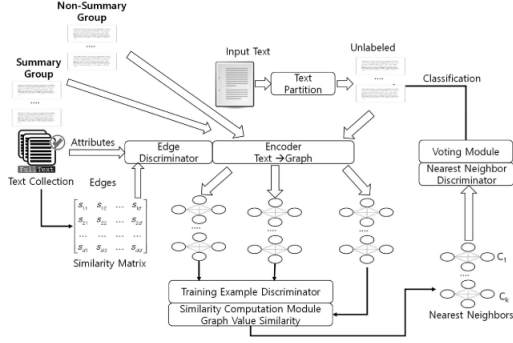


Figure 7. System Architecture

or non-summary, and there are two groups of paragraphs in the text: summary group and non-summary group. The paragraph which are classified into summary are extracted as the summary of the input text in this system.

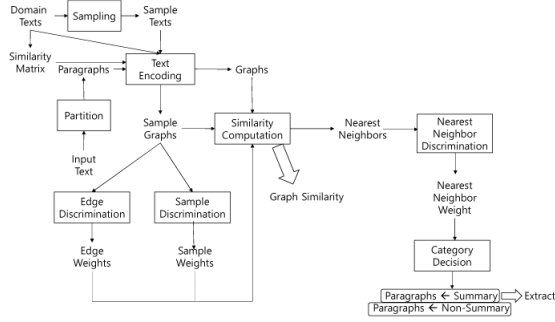


Figure 8. System Flow

Let us make some remarks on the proposed system which is illustrated in Figure 7 as its architecture. The text summarization is defined as the binary classification where each paragraph is classified into summary or non-summary. Paragraphs are encoded into graphs instead of numerical vectors, and they are classified directly. Ones which are classified with summary are selected as the summary of the input text. We need to add the text classification module for predicting the domain of input text automatically.

D. Connection with Text Classification

This section is concerned with the connection of the text summarization with the text classification. In the previous section, we mentioned the text summarization system as an instance of domain dependent classification. The domain is automatically decided for nominating a classifier by connecting the keyword with the text classification. The KNN algorithm which was described in Section II-B is used for implementing the text classification. This section is intended to describe the connection of the text summarization system with the text classification for automating the decision of the domain.

The system architecture of the text categorization system which consists of the encoding module, the similarity computation module, and the voting module is illustrated in Figure 9. The encoding module is for encoding texts into graphs by the process which is described in Section II-A. The similarity computation module is for computing the similarities between graphs which represent a novice text and a sample text and selecting ones with their highest similarities as the nearest neighbors. The voting module is for deciding the label of the novice item by voting the labels of the nearest neighbors. This system used for automating deciding the domain of the input text.

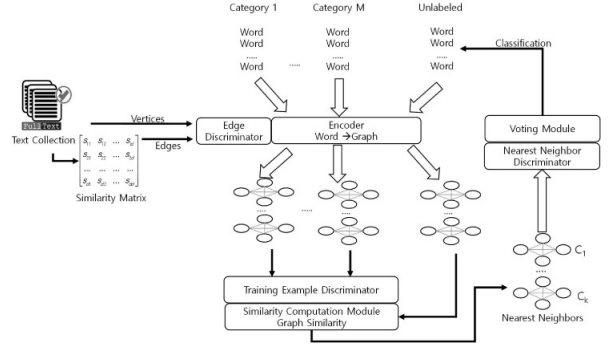


Figure 9. Text Categorization System

The connection of the text summarization with the text classification is illustrated in Figure 10. The texts each of which is labeled with its own domain are prepared as the sample texts, and the approach to the text classification is trained with them. A domain is assigned to a novice text by classifying it into a domain automatically. The novice text is provided with its domain to the text summarization system which is described in Section II-C. The role of the text classification system is to nominate the classifier which corresponds to the domain for the text summarization, automatically.

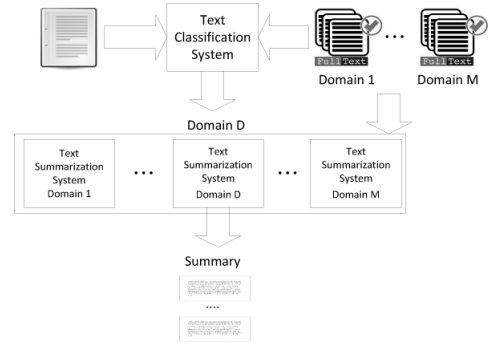


Figure 10. Text Summarization System connected with Text categorization

The process of executing the text summarization, connecting it with the text classification is illustrated in Figure

11. The sample texts each of which is labeled with its own domain are prepared, and the sample paragraphs each of which is labeled with summary or non-summary are prepared in each domain. A novice text is classified into one among the predefined domains, and the classifier which corresponds to the domain is nominated. The novice text is partitioned into a list of paragraphs, and each paragraph is classified into summary or non-summary. The paragraphs which are labeled with summary is the final output from this system; the domain which is assigned to the input text is the guide for selecting a classifier.

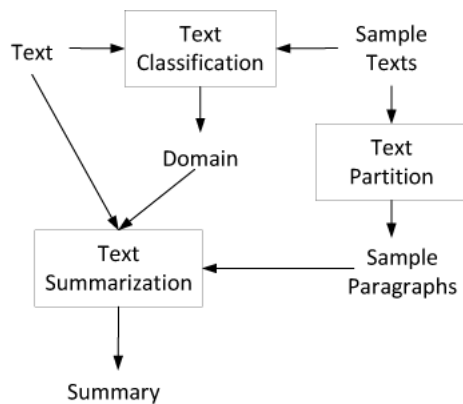


Figure 11. System Flow of Text Summarization connected with Text Categorization

Let us make some remarks on the connection of the text summarization with the text classification. The text classification is applied to the automatic decision of the input text domain. The role of text classification is to decide the domain for nominating a classifier, and the role of the text summarization is to extract important paragraphs as the summary. In the execution flow, the input text is classified into a domain, it is partitioned into a list of paragraphs, and each paragraph is classified into summary or non-summary. We consider connecting the text classification as the main task the text summarization as the opposite to what is proposed in this research.

REFERENCES

- [1] T. Jo, "Graph based KNN for Optimizing Index of News Articles", 53-62, Journal of Multimedia Information System, 3, 2016.
- [2] T. Jo, "Comparing Graph based K Nearest Neighbor with Traditional Version in Word Categorization in NewsPage.com, 12-18, International Journal of Advanced Social Sciences, 1, 2018.
- [3] T. Jo, "Validation of Graph based K Nearest Neighbor for Summarizing News Articles", 5-8, The Proceedings of International Conference on Green and Human Information Technology Part II, 2019.